Energy and Switch Area Optimizations for FPGA Global Routing Architectures

YI ZHU, YUANFANG HU, MICHAEL B. TAYLOR, and CHUNG-KUAN CHENG
University of California, San Diego

Low energy and small switch area usage are two important design objectives in FPGA global routing architecture design. This article presents an improved MCF model based CAD flow that performs aggressive optimizations, such as topology and wire style optimization, to reduce the energy and switch area of FPGA global routing architectures. The experiments show that when compared to traditional mesh architecture, the optimized FPGA routing architectures achieve up to 10% to 15% energy savings and up to 20% switch area savings in average for a set of seven benchmark circuits.

Categories and Subject Descriptors: B.7.1 [**Integrated Circuits**]: Types and Design Styles—*Gate arrays*; B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids

General Terms: Algorithms, Design, Performance

Additional Key Words and Phrases: FPGA, global routing, low power

ACM Reference Format:

Zhu, Y., Hu, Y., Taylor, M. B., and Cheng, C.-K. 2009. Energy and switch area optimizations for FPGA global routing architectures. ACM Trans. Des. Autom. Elect. Syst., 14, 1, Article 13 (January 2009), 25 pages, DOI = 10.1145/1455229.1455242. http://doi.acm.org/10.1145/1455229.1455242.

1. INTRODUCTION

Low energy and small switch area usage are two important design objectives in FPGA global routing architecture design. In contrast to ASICs, which connect logic using fixed metal wires, FPGAs employ programmable switches. Although these programmable switches bring flexibility to FPGAs, they lead to greater energy and on-chip area usage, making FPGAs less favorable in energy-critical applications such as portable devices [Betz et al. 1999]. In this article, we study how to effectively reduce energy and switch area usage of FPGA routing architectures through a multicommodity flow (MCF) model based CAD flow.

Topology optimization can effectively reduce energy and switch area of FPGA routing architecture. Traditionally, people adopt a mesh topology for FPGA global routing architectures due to its simplicity. However, as feature sizes

Authors' address: 9500 Gilman Dr., La Jolla, CA 92093; email: {y2zhu,yhu,mbtaylor,kuan}@cs.ucsd.edu.

shrink and die sizes grow, interconnect energy consumption can become a serious issue for these traditional mesh architectures [Poon et al. 2002]. Furthermore, as pointed out in DeHon and Rubin [2004], mesh routing schemes suffer from unscalable switching area requirements. Therefore, more complex topologies have the potential to improve the energy and area efficiency of FPGA routing architectures. Segmentation distribution is one of the effective techniques to explore FPGA topology space. Brown et al. investigated differentiation segmentation distributions in order to optimize the speed and area [Brown et al. 1996, 1998; Khellah et al. 1994]. Chow et al. [1999] performed a similar study on the impact of segmentation distributions on circuit routability. Lee et al. [2003] explored more versatile wire segmentations and richer connections of FPGA routing architecture to improve routability and reduce delay.

Compared with topology optimization, which has been widely studied for many years by lots of researchers, wire style optimization emerges only in recent years as a result of rapid advances in signaling interconnect technologies. A few works explored the introduction of multiple signaling technologies to low power network-on-chip (NoC) design [Hu et al. 2005], as well as communication latency constrained, low power NoC design [Hu et al. 2006]. Other work studied bus-based connections to improve FPGA switch area density [Ye et al. 2003; Ye and Rose 2005].

We integrate both topology and wire style optimization in our optimization framework to reduce energy usage and switch area of FPGA routing architecture. Our methodology is based on two MCF models: the first synthesizes the optimized FPGA global routing architecture with topology and wire style optimizations, while the second evaluates the optimized FPGA architecture over a set of benchmark circuits. MCF models have been studied for many years in global routing research, and have proven to be an effective tool in global routing optimization. Carden et al. [Carden and Cheng 1991; Carden et al. 1996] routed multiterminal netlists using the approximation MCF algorithm by Shahrokhi and Matula [1990]. Garg and Konemann [1998] had breakthrough improvements of the algorithm, and Albrecht [2000] applied the new algorithm to render MCF based global routing practical for full chip design. Albrecht further extended the MCF model for more optimizations in congestion and timing-driven global routing, including buffer insertion, pin assignment, and buffer wire sizing [Albrecht et al. 2007]. The basic approach of these previous works is similar to ours. However, we introduce multiple wire styles into the MCF models, and optimally assign the capacities for these wire styles for interconnections, which is shown to largely improve the energy and switch area in our FPGA global routing architecture design.

The rest of the article is organized as follows: Section 2 briefly explains the ideas of topology and wire style optimizations. Section 3 first describes our improved CAD flow to generate optimized FPGA global routing architectures, and then explains in detail the core components in the design flow, that is, representative netlist generation and two MCF models. Section 4 describes in detail our approximation MCF algorithms and interval estimation technique to speed up the process. Section 5 presents the experimental results. We summarize our study in Section 6.

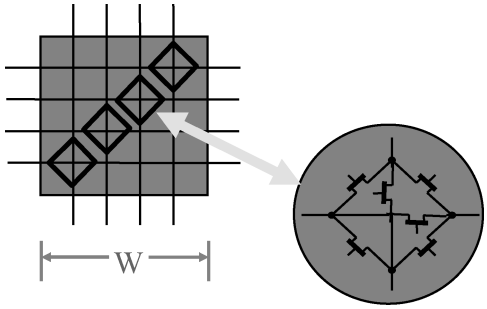

Fig. 1. A switch in FPGA routing architectures.

2. TOPOLOGY AND WIRE STYLE OPTIMIZATIONS

In our work, we perform two types of optimizations to reduce the energy and switch area of FPGA routing architectures. They are wire style optimization and topology optimization.

Wire style optimization studies how to assign various wiring technologies to wire segments in FPGA routing architectures. Recent advances in signaling interconnect technologies, such as wave-pipelined RC wires with repeated buffers, low-swing differential pairs, and on-chip transmission lines, provide us various wiring schemes to optimize aggressively. These technologies, along with traditional minimal separated RC wires, display different tradeoffs between wire resources and power consumption. For example, on-chip transmission lines usually consume less energy, but with larger routing area. On the other hand, traditional minimal separated RC wires occupy less space, but have worse energy efficiency. Therefore, when there is extra on-chip routing space existing, we have room to perform wire style optimization.

Topology optimization is a generalization of segmentation distribution technique [Brown et al. 1996], which introduces wire segments of various lengths to FPGA routing architecture. It has a profound impact on energy consumption when combined together with wire style optimization, because the choice of routing topology impacts not only the routability but also the utilization of available wiring technologies. For example, for on-chip transmission lines, due to the overhead of transmitter and receiver circuits, brings energy and speed benefits only for long wires. Thus, a topology with long links can make better use of such advanced wiring technologies. Topology optimization also has an impact on switch area efficiency. Figure 1 demonstrates a switch box in an FPGA routing architecture, where each switch block on the intersection point is composed of six switch circuits. The switch area of a switch box is determined by the number of switches inside a switch box, which is proportional to the number of input/output wire tracks W. Since topology optimization is an important factor to affect W, it has important impact on switch area efficiency.

To perform topology optimization, we first generate a set of candidate topologies and put them in the topology library. The topologies of our optimized FPGA routing architectures are selected from the topology library. The library can be easily expanded by importing valuable candidate topologies.

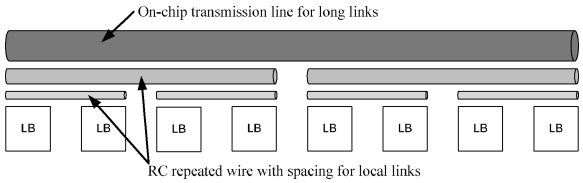

Fig. 2. Topology and wire style optimizations in FPGA routing architectures.

Topology library generation is critical to the success of our FPGA routing architecture optimization approach. Even after clustering the look-up tables (LUTs) into larger logic blocks, there are still a huge number of possible topologies. For example, for an FPGA with 10×10 logic blocks, each row or column has $2^{C(10,2)} = 2^{45}$ different connections, and the whole FPGA chip has $(2^{45})^{20} = 2^{900}$ different connections. It is impossible to explore them exhaustively with the current computation technology.

To reduce the size of topology library and only keep the most valuable and promising topologies, we make a few strategic assumptions. First, we assume all wire segments have lengths of power of two; that is, there are only wires in lengths of 1, 2, 4, 8, etc. Second, in each row or column, wire segments should repeat themselves consecutively along the whole routing channel. Third, all rows and columns should have identical connections. The reasoning of these assumptions is that: at the stage of FPGA global routing architecture design, the target applications are still unknown, therefore it is reasonable to design relatively regular and symmetric topologies to fit potential applications. Based on the above assumption, we exhaustively generate all qualified candidate topologies.

Figure 2 shows an example of topology and wire style optimizations in FPGA routing architectures. The logic blocks (LB) are connected by wire segments of various lengths, and different wire segments can be implemented with different wire styles.

3. DESIGN METHODOLOGY

In our work, an MCF model based optimization framework, which integrates both topology and wire style optimizations, is introduced to an existing CAD flow. The optimization framework takes each candidate topology and available wire styles as inputs, and produces optimized capacities and wire style assignments for wire segments in FPGA routing architecture. By repeating this process for all candidate topologies in topology library, we can obtain the best topology with wire style optimization as our optimized FPGA routing architecture.

In the following sections, we first describe our improved CAD flow, then explain the major components in the design flow.

3.1 An Improved CAD Flow

Figure 3 shows our improved CAD flow. The inputs are a set of benchmark circuits. The first a few steps, which are unshadowed in Figure 3, are the traditional flow used in the literature:

(1) First, we use SIS [Sentovich and Al 1992] to perform technology independent logic optimization on each of the circuit. The tool could

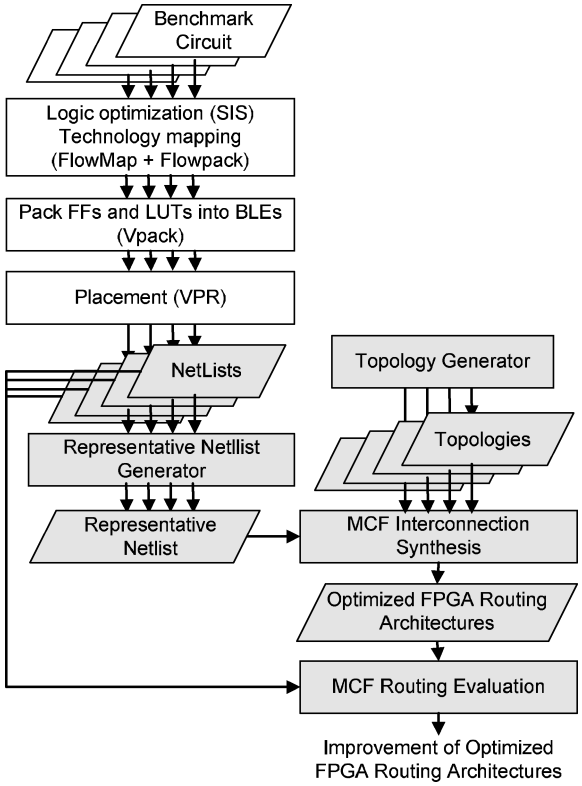

Fig. 3. An improved CAD flow for FPGA routing architecture optimization.

produce optimized netlists while preserving sequential input/output behavior [Sentovich and Al 1992].

(2) Next, the optimized circuits are technology-mapped by FlowMap [Cong and Ding 1994] into four-input lookup tables (4-LUTs). Then Flowpack [Cong and Ding 1994] is used to optimize the mapping and reduce the number of LUTs required.

(3) Thirdly, we use VPack [Betz and Rose 1997] to pack these 4-LUTs into larger logic blocks. VPack takes as input a technology-mapped netlist of LUTs and flip flops, and output a netlist composed of more complex logic blocks.

(4) The resulting netlists are then fed into VPR [Betz and Rose 1997], and are placed on the FPGA chip. The placement results provide the traffic patterns used by our representative netlist generator, which will be described in the next.

The following shadowed steps in Figure 3 are our improved parts. We have a netlist generator to generate the representative netlist by extracting the traffic characteristics of the input benchmark circuits, which are provided by the VPR placement results.

The representative netlist reflects the traffic distribution of the benchmark circuits, hence effectively guides the design of FPGA routing architectures to fit for the largest class of the benchmarks. Meanwhile, we use a topology generator to generate a set of candidate FPGA topologies. The representative netlist and the candidate FPGA topologies are then fed into the MCF interconnection synthesis tool, which models the FPGA routing optimization problems with specific objectives, such as energy or switch area usage. The output of the MCF formulations will be the optimized FPGA global routing architectures with topology and wire style optimization. In the last step, the benchmark circuits are fed to the optimized routing architectures, and an MCF routing evaluation model is used to evaluate the actual improvement.

Compared to the traditional CAD flow in FPGA design, our improved design flow is able to automatically generate candidate global routing architectures. This largely increases the flexibility of global routing architectures and explores a much larger design space.

In our improved CAD flow, there are two major components. One is a netlist generator to generate representative netlist, the other core component includes two MCF models, MCF interconnection synthesis and MCF routing evaluation. We describe each of them in the following subsections.

3.2 Representative Netlist Generation

To achieve the best benefits of our FPGA routing optimizations, we need to have a good understanding of the nature of communications in our applications. The performance of both topology and wire style optimizations largely depends on the underlying communication pattern. For example, routing architectures with long distance transmission lines may be able to effectively reduce the energy consumption of the applications which have largely global communication, but it may bring negative effects for those applications where most of communications are local.

Therefore, we generate a representative netlist from a set of FPGA benchmark circuits. The generated representative netlist should catch the characteristics of the benchmark circuits. The representative netlist generation is based on the statistical analysis of the candidate application circuits. Three sets of key parameters need to be determined. First, how many nets should the netlist have? Second, what is the size, for example, the number of pins, of each net? Finally, what are the pin locations of each net?

Hutton et al. [1998] describe in detail how to characterize the benchmark FPGA circuits. They raised several important parameters to describe the circuit properties, including size, shape, edge length distribution and fanout distribution, etc. They then proposed an algorithm to generate parameterized synthetic circuits. Here, however, we have a bit different purpose here—we would like to have only one large circuit that is able to represent the characteristics of all benchmarks. Therefore, we employ the following algorithm to generate the representative netlist, which is a bit different from that in Hutton et al. [1998].

We set the size of the representative netlist to be the maximum netlist size among all the benchmark circuits. Because a netlist with larger size usually

requires more routing capacity to route, it is intuitive to assign the representative netlist with the maximum routing capacity requirements among the benchmarks, so that the optimized FPGA routing architectures can be reconfigurable to accommodate all benchmark circuits.

To determine the size of each net, we first count the net size of all the benchmark circuits, and calculate their distribution. Then we design the size of each net in the representative netlist to match this distribution pattern. For example, if 5% of nets in benchmark circuits have number of pins in the range from 30 to 35, and if the size of our representative netlist is 1000, then we should evenly distribute 50 nets with pins in that range.

Finally, we need to determine the pin locations of each net. Random generation of pin locations may lose the intrinsic communication patterns of the benchmark circuits. Therefore, we analyze the distribution of the frequency of each pin in the candidate circuits, and generate a corresponding "pin pool" with frequency distribution for each pin in the pool. Then for each net, we pick pins from the pin pool according to their frequency function. In this way, we preserve part of the communication patterns of the benchmark circuits.

We determine the distance among pins by a *geometry distribution function*, which closely relates to most industrial circuit configurations [Shyu et al. 2000]. The function is defined as the probability of the distance between two pins which decreases exponentially with increasing distance; that is,

$$P(k) = p(1-p)^k, k = 1, 2, \ldots, \tag{1}$$

where k is the distance between two pins, p is the probability of links with distance 1, and $P(k)$ is the probability of links with distance k. The other approach to predict the wire length is based on Rent's rule [Donath 1981]. Christie and Stroobandt [2000] gave a comprehensive review on this method, and Dambre et al. [2004] further improved the model later. In their model, the probability is a polynomial function of the distance, instead of exponential. We adopt the first approach in our experiments, because Hutton et al. [1998] computed the distances among the pins in MCNC benchmarks, which are used in our experiments, and they showed that their distribution is similar to an exponential function.

Notice that, in our work, we simplify the problem by having the generated representative netlist connect switch boxes instead of logic blocks, by distributing the pins of nets on logic blocks to the adjacent clockwise switch boxes. By inserting logic boxes as extra nodes into our MCF models, we can easily improve the accuracy of our methodology, but with the price of higher computational cost.

3.3 MCF Interconnection Synthesis and Routing Evaluation

As shown in Figure 3, at the core of our optimization framework are two MCF models. The first is the MCF interconnection synthesis model, which generates the optimized FPGA routing architectures with topology and wire style optimizations for representative netlist. The other model is MCF routing evaluation, which evaluates the actual performance of these optimized FPGA routing architectures for the target set of benchmark circuits.

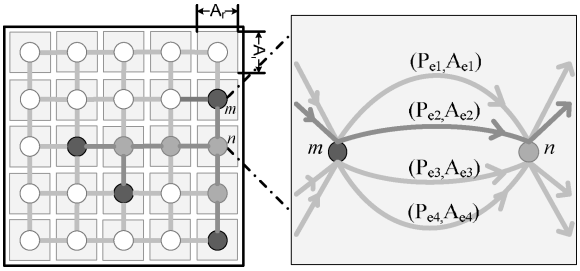

Fig. 4. An improved CAD flow for FPGA routing architecture optimization.

The major difference between these two MCF models lies in their constraints. In MCF interconnection synthesis, the capacity of each routing channel is unknown, hence it regards on-chip area resources of the routing channel as its constraints, and generates optimized capacities for each routing channel. While the constraints for MCF routing evaluation are these output routing capacities.

The following subsections describe these two models in detail. First, we show how to integrate the wire style optimization into the MCF models. Then, we present the formulations for each MCF model with various design objectives in mind. This demonstrates the flexibility of our methodology to be adapted to a wide range of FPGA routing architecture optimizations. Finally, we briefly describe the algorithms that can efficiently solve these MCF models.

3.3.1 *Integration of Wire Style Optimization.* Assume an FPGA chip with $n \times n$ logic blocks. These logic blocks communicate with each other through $n \times n$ switch boxes at the intersection of the channels. A topology is defined as a bidirected graph $G = (V, E)$, where, each node $v_i \in V$ represents a switch box, and each edge $e_{i,j} \in E$ represents routing tracks between switch boxes i and j. These wire tracks can be implemented with multiple wire styles. Assume there are k nets. For each net i, its communication demand is $d_i = 1$. Let t_i be the set of paths on Steiner trees to connect net i, and let $\mathbf{T} := \cup_i t_i$. Variable $f(t)$ denotes the amount of flow along Steiner tree t, for every $t \in \mathbf{T}$.

Figure 4 demonstrates an example on how to integrate multiple wire styles into MCF models. In a mesh architecture, we have a net of 4 pins (black nodes) to be routed. We connect these pins using a minimum Steiner tree (grey nodes are Steiner nodes), as shown in dark lines in left side of the figure. Then we use multiple edges to represent available wire styles, as shown in right side of the figure. For link (m, n), there are 4 edges from node m to node n, which represents 4 types of candidate wire styles. A pair (P_e, A_e) is associated with each edge e. P_e is per bit energy on edge e. A_e is the wire pitch. If there is flow going through edge 2 (as shown in the dark line), it means wire style 2 is selected for link (m, n), and the capacity of edge 2 equals to the amount of the flow. Therefore, if we solve the MCF formulations and get the flow distribution, we can obtain the optimized global routing architecture with wire style optimization.

Here we assume unidirectional routing when performing the synthesis, because directional routing and single-driver wiring, where the wires are dedicated for one direction transmissions, have been shown to be superior to the

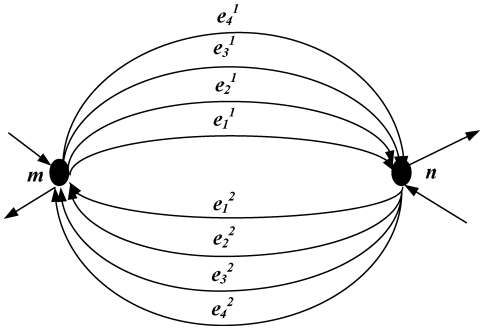

Fig. 5. Graph model for directional routing.

traditional bidirectional routing [Lemieux et al. 2004]. As we have mentioned, the topology graph G is bidirected, that is, between a pair of nodes m and n, there are two sets of edges as shown in Figure 5. One set of edges $\{e_1^1, e_2^1, e_3^1, e_4^1\}$ are directed from m to n, and the other set $\{e_1^2, e_2^2, e_3^2, e_4^2\}$ are opposite. Each set contains 4 edges for 4 different wire styles. Note that the flow that goes through each edge is independent, which implies that the number of wires routed in two directions could be different. This increases the flexibility of routing and the utilization of the resources.

3.3.2 *MCF Interconnection Synthesis.* Different FPGA optimization problems correspond to different MCF interconnection synthesis formulations. MCF model has the flexibility to adapt to various design objectives. In our work, we study three types of optimization problems, focusing on the energy optimization, the switch area optimization, and their cooptimization. These optimization problems are important and of the interests in modern FPGA routing architecture design.

For the first problem, we optimize the energy of the FPGA routing architecture. To estimate energy, for each edge e, we assume that P_e represents bit energy on link e and the corresponding switch box.

$$P_e = P_w + P_{sb},$$

where P_w and P_{sb} are bit energy on interconnects and switch box, respectively. When a flow of amount f goes through the edge and the corresponding switch box, the total energy is $P = P_e \cdot f$

P_{sb} can be estimated by

$$P_{sb} = P_s \cdot N_s,$$

where P_s is energy for a single switch in a switch box, and N_s is the total number of switches in a switch box. Assume F_s is number of switches connected to each wire entering a switch box, and f is the amount of flow go through a switch box, we have:

$$N_s = 1/2 \cdot F_s \cdot f,$$

The following is the formulation for MCF synthesis on energy optimization. The objective is to minimize the total energy of the routing architecture, which

is the sum of per-bit energy on all routing tracks (as in Equation (2)). We have two constraints. The routability constraint (3) requires that all the nets in the representative netlist should be routable, while the routing area constraint (4) ensures that when we route the nets, the routing area usage cannot exceed the available on-chip area resources on the vertical or horizontal dimension A_r.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot P_e \tag{2}$$

$$s.t. \quad \forall 1 \leq j \leq k : \sum_{t \in T_j} f(t) \geq 1 \tag{3}$$

$$\forall q : \sum_{e \in Grid(q)} A_e \cdot \sum_{t : e \in t} f(t) \leq A_r \tag{4}$$

$$\forall t : f(t) \geq 0. \tag{5}$$

The outputs of the MCF synthesis model are the optimized FPGA global routing architectures with expected design objectives, in this case, expected energy on routing architectures. Notice that the variables in the formulations are $f(t)$. After we solve the MCF formulations, we can obtain the optimized capacity for each edge by calculating the accumulated $f(t)$ on that edge. In this way, we have the optimized FPGA routing architecture for a certain topology. Then we can repeat this process for each candidate topology and generate the optimized FPGA routing architectures with topology and wire style optimizations.

In the second case, we optimize the total switch area of switch boxes. Since the switch area is proportional to the number of switches, we try to minimize the total number of switches in switch boxes as our design objective. The constraints of this problem are exactly the same as those of the first case, that is, the routability and routing area constraints. Therefore, we omit the constraints part of the formulations here, and only give the objective function as follows, in which energy parameters P_e are simply replaced by switch quantity parameters N_s.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot N_s. \tag{6}$$

Furthermore, such MCF synthesis model can be easily applied to study the tradeoffs between multiple design factors. In our third case, we study the switch area constrained energy optimization problem, which means we optimize the energy of the FPGA routing architecture, while at the same time satisfying all requirements on total switch area usage. Compared with the first energy optimization problem, this problem has one more switch area constraint (7), where A_s is the given switch area budget. The objective function and the other constraints are exactly the same as formulation (2), (3), and (4).

$$\sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e \in t} f(t) \cdot N_s \leq A_s. \tag{7}$$

3.3.3 *MCF Routing Evaluation.* As we explained earlier, MCF routing evaluation model differs from MCF interconnection synthesis model only on that one of its constraints is routing channel capacity instead of on-chip routing area resources. Take energy optimization problem as an example, its MCF routing evaluation formulations are as follows, where $c(e)$ represents the capacity of edge e.

$$Min: \quad \sum_{j=1}^{k} \sum_{t \in T_j} \sum_{e} f(t) \cdot P_e \tag{8}$$

$$s.t. \quad \forall 1 \le j \le k : \sum_{t \in T_j} f(t) \ge 1 \tag{9}$$

$$\forall e : \sum_{t:e \in t} f(t) \le c(e) \tag{10}$$

$$\forall t : f(t) \ge 0. \tag{11}$$

The outputs of MCF routing evaluation model are the actual design results, such as total energy, for each of benchmark circuits. This evaluation process verifies the effectiveness of MCF interconnection synthesis model.

4. APPROXIMATION ALGORITHMS FOR MCF MODELS

We adopt the polynomial time approximation algorithms to quickly solve the above two MCF models. Both algorithms are based on linear programming (LP) primal-dual theory and can obtain $(1 + \epsilon)$ optimal solution in polynomial time. The core idea in the algorithms is to iteratively perform minimum Steiner tree algorithm to update dual and primal solutions, so that the gap between them can finally be reduced below the error bound. For MCF routing evaluation model, the algorithm is the same as that in Albrecht [2000]. For MCF interconnection synthesis model, the algorithm is slightly different, as presented in the following. In addition, we propose the interval estimation technique to speed up the process.

4.1 Baseline Algorithm

We describe the approximation algorithm for MCF interconnection synthesis model for energy optimization. The same algorithm can be easily applied to the other optimizations, such as switch area optimization, and switch area constrained energy optimization, etc.

The algorithm has two steps. Let P be a target for the total energy, we first solve a corresponding maximum concurrent flow problem, which finds the largest relative congestion λ such that the total energy of the multicommodity flow is constrained by P. In the second step, we perform binary search over P till P is as small as possible and λ is at least one. In practice, the total energy in the final solution is only slightly higher compared to the minimum energy if each Steiner tree is as short as possible ignoring capacities. This gives a good estimate for P.

The maximum concurrent flow problem of MCF interconnection synthesis for energy optimization is as follows (corresponding to Equations (1)–(4)).

$$\begin{aligned} Max: \quad & \lambda \\ s.t. \quad & \forall j: \sum_{t \in T_j} f(t) \geq \lambda \\ & \forall q: \sum_{e \in Grid(q)} A_e \sum_{t: e \in t} f(t) \leq A_r \\ & \sum_{i=1}^{k} \sum_{t \in T_i} \sum_{e \in t} f(t) \cdot P_e \leq PW \\ & \forall t: f(t) \geq 0. \end{aligned}$$

The dual of this linear program is given as follows. It has variable X_q for area constraint, Z_j corresponds to each netlist, and ϕ for energy constraint.

$$\begin{aligned} Min: \quad & A_r \sum_{q=1}^{n} X_q + PW \cdot \phi \\ s.t. \quad & \forall j, \forall t \in T_j: \sum_{e \in t} A_e \sum_{e \in Grid(q)} X_q + \sum_{e \in t} P_e \cdot \phi \geq Z_j \\ & \sum_{j=1}^{k} Z_j \geq 1 \\ & \forall q: X_q \geq 0, \forall j: Z_j \geq 0. \end{aligned}$$

By the LP duality theory, any feasible solution of the dual linear program provides a lower bound on the optimum solution for the primal problem. To calculate dual values, we define edge length as:

$$l(e) := A_e \sum_{q: e \in Grid(q)} X_q + PW \cdot \phi. \tag{12}$$

So the dual is equivalent to:

$$Min: \frac{A_r \sum_{q=1}^{n} X_q + PW \cdot \phi}{\sum_{j=1}^{k} d_j \cdot dist(j)}, \tag{13}$$

where $dist(j)$ is the minimum Steiner tree for net j under the length function $l(e)$. The algorithm is given as follows.

Algorithm 1 proceeds in phases and each phase is composed of k iterations. In iteration j of the i^{th} phase we route net j in a sequence of steps. In each step, a minimum Steiner tree t is computed using the current length function. We adopt the minimum Steiner tree algorithm from Mehlhorn [1988], which has complexity of $O(|E|+|V|log|V|)$. The dual variables X_q, ϕ and edge length $l(e)$ are updated in steps 11-13. As proven in Albrecht [2000], Algorithm 1 can converge in $O(1/\delta^2 \lambda \ln m)$ phases, where m is total number of edges in the graph G.

In the algorithm, it is worth noting that the dual variables X_q are associated with a set of edges instead of a single edge, therefore we need to apply formula (12) to further compute the edge lengths. This is from the intrinsic spirit of the

Algorithm 1. $(1-\delta)$ Maximum Concurrent Flow Algorithm

Input: graph G, energy budget PW, threshold δ
Output: $(1-\delta)$ optimal maximum concurrent value λ
$\forall q$, set $f(e) \leftarrow 0$, $X_q \leftarrow X_0$, $\phi \leftarrow X_0$
$l(e) \leftarrow A_e \sum_{q:e \in Grid(q)} X_q + PW \cdot \phi$
while $A_r \sum_{q=1}^{n} X_q + PW \cdot \phi \leq 1$ **do**
  for each net j **do**
    $rd_j \leftarrow 1$
    while $rd_j > 0$ **do**
      Find a minimal Steiner tree $t \in T_j$ for net j with respect to length $l(e)$, route flow f
      $f(e) \leftarrow f(e) + f, \forall e \in t$
      $X_q \leftarrow X_q(1 + \frac{\delta}{3} \cdot \frac{\sum_{e \in Grid(q)} A_e f(e)}{A_r})$
      $\phi \leftarrow \phi(1 + \frac{\delta}{3} \cdot \frac{\sum power}{PW})$
      $l(e) \leftarrow A_e \sum_{q:e \in Grid(q)} X_q + PW \cdot \phi$
      $rd_j \leftarrow rd_j - f$
    end while
  end for
  $D \leftarrow \frac{A_r \sum_{q=1}^{n} X_q}{\sum_{j=1}^{k} \cdot dist(j)}$
end while
return λ

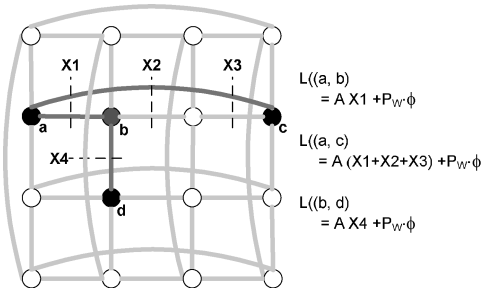

Fig. 6. Length function on edge.

dual variable updating scheme: the dual variables reflect the congestion level of the edge or grid, consequently we always update it using the ratio of the flow versus the total available resource. Sometimes this results complicated cases. Refer to Figure 6, consider a Steiner tree consisting of edges (a, b), (a, c) and (b, d), the lengths should be updated accordingly as shown in Figure 6.

Once we have the solution λ for the maximum concurrent flow problem, in the second step we find the optimized energy that satisfying $\lambda \geq 1$ by recursive binary search, as shown in Algorithm 2, where we use λ_{max} to denote the concurrent value without energy constraint; that is, $PW = \infty$.

4.2 Interval Estimation

While Algorithm 2 needs to obtain MCF solutions with $(1+\epsilon)$ optimal energy values, Algorithm 1 returns the $(1-\delta)$ optimal concurrent flow. Therefore the

Algorithm 2. Energy Optimization MCF Interconnection Synthesis Algorithm

```
1: Input: graph G, threshold ε
2: Output: (1 + ε) optimal energy
3: set λ_max ← mcf(G, ∞)
4: set lower bound lb ← 0
5: upper bound ub ← total energy under λ_max
6: while (ub − lb)/ub > ε do
7:    λ ← mcf(G, (lb + ub)/2)
8:    if λ ≥ 1 then
9:       ub ← (lb + ub)/2
10:      else lb ← (lb + ub)/2
11:   end if
12: end while
13: Output ub
```

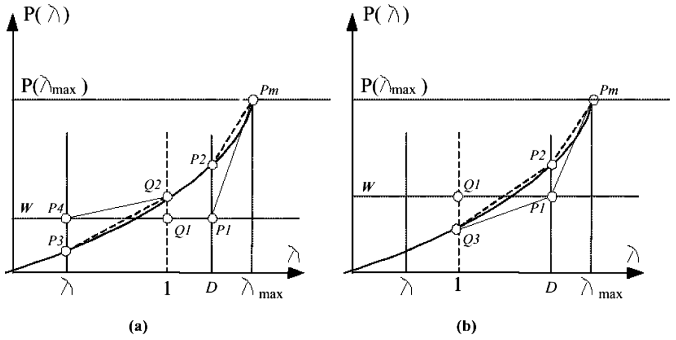

Fig. 7. Interval estimation.

values of ϵ and δ are associated "pseudo polynomially": δ has to be determined by both the value of ϵ and the unit edge cost P_e, which leads to extremely slow convergence in some pathological cases.

We propose a heuristic interval estimation technique to speedup the process. The idea is to estimate the new lower bound lb' and upper bound ub' while performing the approximation algorithms, and break once $ub' - lb' \leq (ub - lb)/2$ in each step of the binary search scheme.

We define a function monotonically increasing $P(\lambda)$, where λ is the concurrent flow and $P(\lambda)$ is the minimum energy under this concurrent flow (therefore $P(1)$ is the target optimal value). The curve is shown in Figure 7. Furthermore, we have the following lemma:

LEMMA: $P(\lambda)$ is a convex function.

PROOF. For a specific λ_1, the minimum energy should be $P(\lambda_1)$; scaling down all the flows by half, the concurrent flow would be $\frac{\lambda_1}{2}$, and the energy is $\frac{P(\lambda_1)}{2}$. On the other hand, when the concurrent flow is $\frac{\lambda_1}{2}$, the minimum energy should be $P(\frac{\lambda_1}{2})$, therefore we have $P(\frac{\lambda_1}{2}) \leq \frac{P(\lambda_1)}{2}$, $\forall \lambda_1 \leq \lambda_{max}$. So the function is convex. □

We use the following theorem to estimate the lower bound lb' and upper bound ub':

Algorithm 3. Modified Energy Optimization MCF Interconnection Synthesis Algorithm

1: **Input:** graph G, threshold ϵ
2: **Output:** $(1+\epsilon)$ optimal power
3: As in Algorithm 2 Steps 3–5
4: **while** $(ub - lb)/ub > \epsilon$ **do**
5: $(lb', ub') \leftarrow mcf(G, (lb + ub)/2)$
6: $lb \leftarrow lb'$; $ub \leftarrow ub'$
7: **end while**
8: Output ub

Algorithm 4. Modified Maximum Concurrent Flow Algorithm

1: **Input:** graph G, energy budget PW, threshold δ
2: **Output:** new lower bound lb' and upper bound ub'
3: As in Algorithm 1 Steps 3–4
4: $lb' \leftarrow lb$; $ub' \leftarrow ub$
5: **repeat**
6: As in Algorithm 1 Steps 6–18
7: $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$
8: $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$
9: **if** $ub' - lb' \leq (ub - lb)/2$ **then**
10: **return** (lb', ub')
11: **end if**
12: **end repeat**

THEOREM 1. *Given a feasible primal value λ and a feasible dual value D under the energy budget PW, we have*

$$PW - s \cdot (D-1) \leq P(1) \leq PW + s \cdot (1-\lambda), \tag{14}$$

where $s = \frac{P(\lambda_{max}) - PW}{\lambda_{max} - D}$. *Hence,* $lb' \leftarrow \max\{lb', PW - s \cdot (D-1)\}$, $ub' \leftarrow \min\{ub', PW + s \cdot (1-\lambda)\}$

We sketch the proof for $P(1) \leq PW + s \cdot (1-\lambda)$ here. Refer to Figure 7 (a), let the lines $x = \lambda$, $x = 1$, $x = D$ and $x = \lambda_{max}$ intersect the function curve at P_3, Q_2, P_2 and P_m, and $x = 1$, $x = 1$ and $x = D$ intersect $y = PW$ at P_4, Q_1 and P_1 respectively (we use x and y to denote the two axes). We then have

$$P(1) = PW + S_{P_4Q_2} \cdot (1-\lambda) \tag{15}$$

where $S_{P_4Q_2}$ is the slope of the line P_4Q_2. And, it is easy to identify that $S_{P_4Q_2} \leq S_{P_3Q_2} \leq S_{P_2P_m} \leq S_{P_1P_m}$, by the property of the convex function. And since $s = S_{P_1P_m}$, we have $P(1) \leq PW + s \cdot (1-\lambda)$. Similarly, $PW - s \cdot (D-1) \leq P(1)$ can be proven by the similar approach, as shown in Figure 7 (b). □

According to Theorem 3, Algorithm 2 and 1 can be improved as Algorithm 3 and Algorithm 4.

5. EXPERIMENTAL RESULTS

In our experiments, we use seven MCNC benchmark circuits [Yang 1991] with moderate sizes. We first perform technology mapping to map these benchmark circuits to 4-LUTs. Then, we pack every 16 4-LUTs into a larger logic block, and finally place these logic blocks on island-style FPGA chip. Table I shows the size

Table I. Size of Representative Netlist of MCNC Benchmark Circuits

	alu4	apex4	diffeq	dsip	ex5p	misex3	tseng
size	11×11	10×10	11×11	11×11	10×10	11×11	10×10
# of nets	621	798	945	593	745	771	788

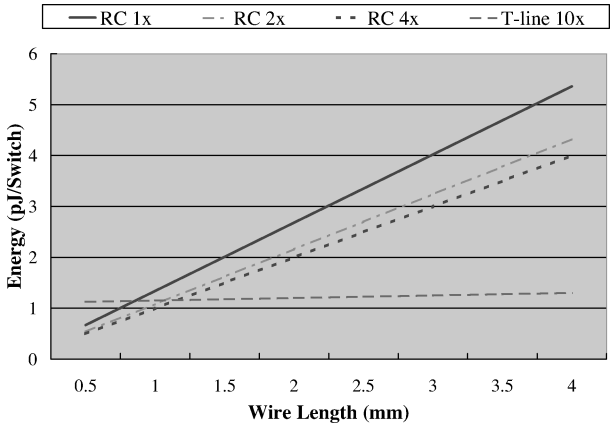

Fig. 8. Energy for four types of wire styles.

of resulting representative netlists of these seven benchmark circuits. Since the size of switch box array ranges from 10×10 to 11×11, the representative netlist is of size 11×11. We set p to be 0.1 in the geometry distribution function $f(k) = p(1-p)^k$ in representative netlist generation, because we observe it best matches the connection nature of our benchmark circuits.

We generate the candidate topologies using the topology generator described in Section 2. In our experiments, we assume the available segment lengths are 1, 2, 4, and 8. Segment of length 1 is mandatory, while other three types of segments are optional. Therefore, the candidate topologies include the base mesh topology, plus others that add arbitrary choices of three extra links. Abiding such assumptions, for FPGA of size 11×11, the total number of generated candidate topologies is 93.

We assume 4 types of candidate wires: RC repeated wires with $1\times$, with $2\times$ and with $4\times$ minimum global pitch as well as transmission lines with $10\times$ minimum pitch. We follow the methodology used in Hu et al. [2006] to estimate the energy: for RC repeated wires, the energy consumption is proportional to the wire length; for transmission line, the model proposed in Chen et al. [2005] is used, which considers an initial setup energy followed by a small linear increment with wire length. The supply voltages, wire geometries and device parameters are from International Technology Roadmap for Semiconductors (ITRS). The setup energy 1.1pJ/bit is added for transmission lines. Figure 8 shows the energy consumption of each wire style under various wire lengths.

In our MCF approximation algorithms, we set error tolerance ϵ to 1%. All of the following experiments are based on $0.18um$ design technology. Since each grid has the same vertical and horizontal dimension, for convenience, we use

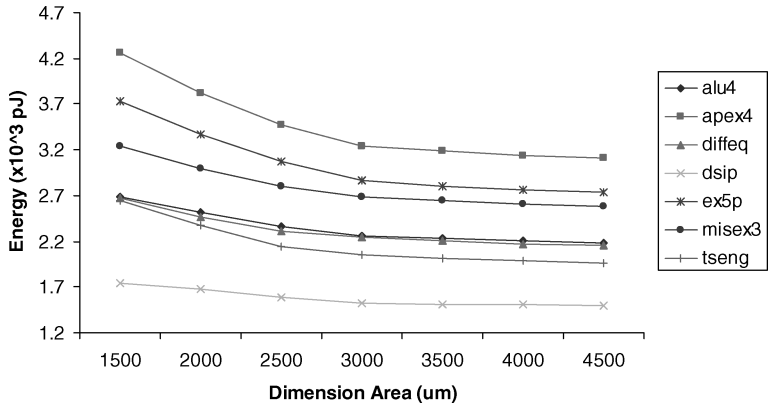

Fig. 9. Energy of benchmark circuits under various routing area constraints.

only the vertical dimension to represent the area budget, therefore the unit of area in our experiments is *um*.

5.1 Energy Optimization

We first demonstrate the impact of the available on-chip routing resources on our energy optimization, and show two examples of the final optimized FPGA routing architectures. Then we compare our optimized routing architectures with traditional mesh architecture to show the improvement from the energy optimization.

5.1.1 *Optimized Energy under Various Routing Area Constraints.* Figure 9 shows the energy of seven benchmark circuits on our optimized FPGA routing architectures. The x-axis is routing area budgets from $1500um$ to $4500um$, which represent the area constraint from tightest to loosest. The y-axis is energy in unit $\times 10^3 pJ$. As area constraints become looser, energy of all benchmark circuits keep decreasing, where circuit *apex4* gains the largest improvement of 27.1% (from 4.26 to 3.11 $\times 10^3 pJ$). The improvements are from both topology and wire style optimizations, since as routing area budgets increase, more energy-efficient but area consuming wires can be adopted in corresponding topologies to reduce the overall energy of FPGA routing architectures.

Figure 10 and Figure 11 shows the detailed topology and wire style assignments in optimized FPGA routing architectures under various routing area constraints. The black blocks represent the switch box arrays, and different wire styles are shown in different colors and line patterns. Figure 10 is when the routing area is $1500um$, and Figure 11 is when the routing area is $4500um$. We observe that in (a) the $1\times$ RC wires are used for most of the connections to save the area usage. Also at the outer regions of the chip, some energy-efficient transmission lines are used to reduce energy, because in those regions the communication flow is not as congested as in the center of the chip, consequently there is room for wire style optimization. In (b), since now we have abundant routing area for wire style optimization, transmission lines are adopted for all the long links, and RC wire with $4\times$ minimum pitch are adopted for those

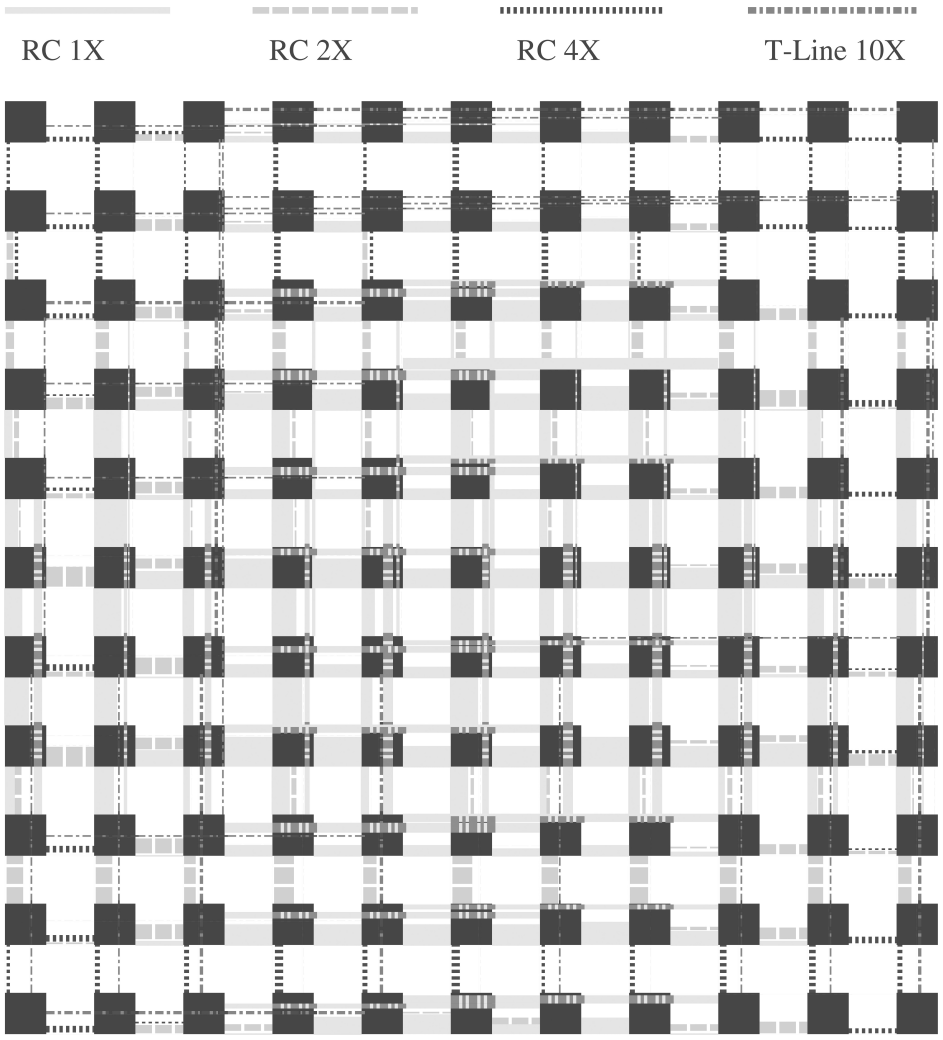

Fig. 10. Optimized FPGA routing architecture when routing area = 1500 um.

short links. As a result, the energy of (b) architecture is 20% less than that of (a) architecture. The bottom of the figure shows the topologies of the corresponding FPGA architectures.

From Figure 10 and Figure 11, we see a clear trend to adopt wider wires as routing area budget increases in order to reduce energy. Also wires at the center of the chip usually are more area-efficient than wires at the outer regions of the chip.

5.1.2 *Energy Improvements over Traditional Mesh Architecture.* We compare the energy of our optimized FPGA routing architectures with that of traditional mesh routing architecture. Figure 12 shows the energy improvement in percentage. In x-axis, each group of bars present the energy under various

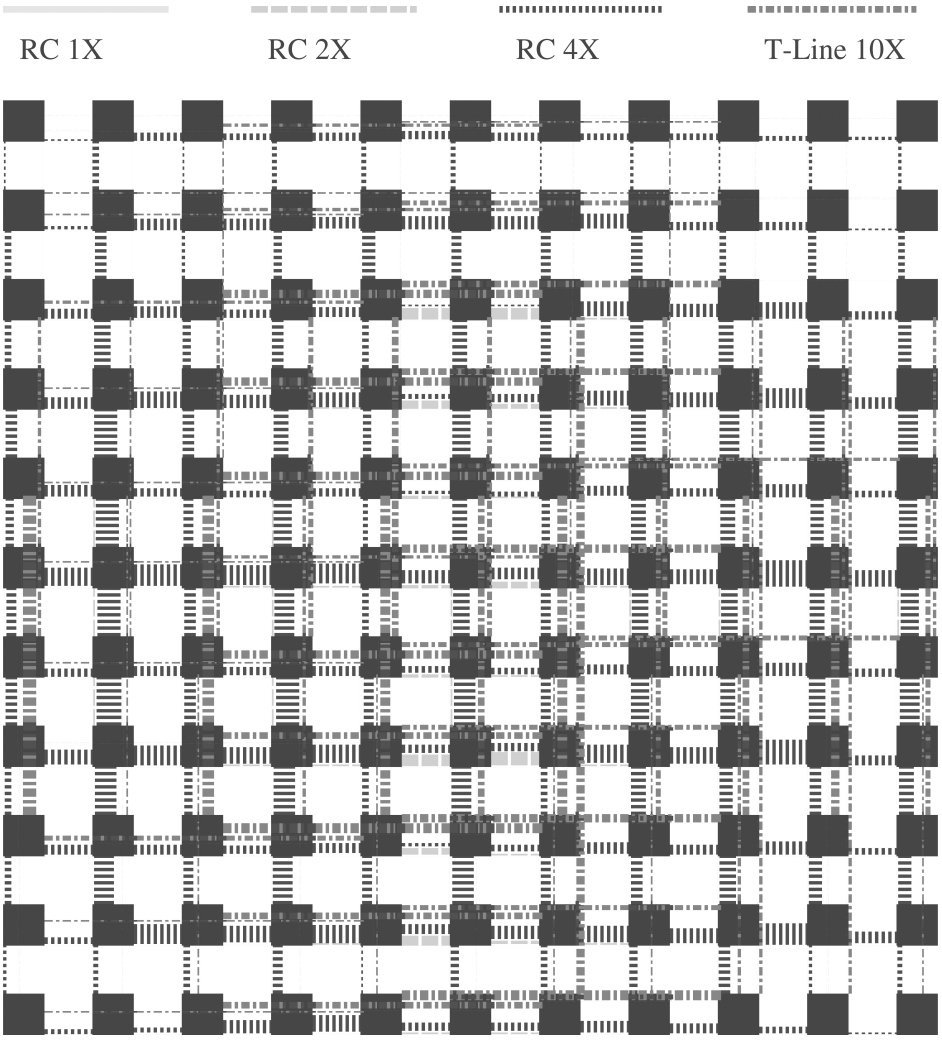

Fig. 11. Optimized FPGA routing architecture when routing area = 4500 um.

area constraints for a certain benchmark circuit. The last bar is energy for representative netlist from MCF interconnection synthesis model, which indicates the estimated improvement of our design.

Circuit *disp* has the smallest improvement, ranging from −3% to 6%; circuit *tseng* has the largest improvement from 5% to 24%. In average, our optimized routing architecture can achieve energy savings from 2% to 15% over mesh architecture. When area budget is small, such as 1500um, our optimized routing architecture has no obvious advantages over traditional mesh architecture, because we do not have enough routing area to adopt better wiring technologies. The major improvement occurs when area budget increase from 1500um to 2500um. Further increasing of routing area budget does not bring too much benefits.

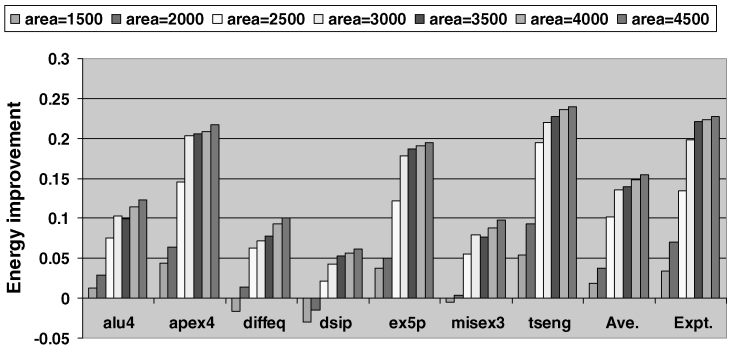

Fig. 12. Improvement of energy of optimized architectures over mesh architecture.

5.1.3 *Energy Comparison Using Power VPR Tool.* We also implement the global routing architectures we find using the FPGA power analysis tool by Poon et al. [2002]. This tool follows the original CAD flow of V-Pack and VPR to pack, place, and route the circuits. In addition, it modifies the V-Pack and VPR packages to embed a power model to analyze the energy of FPGA's using the switching activity information generated by the activity estimator. Readers can refer to the authors' web site [Choy et al.] for details.

Unfortunately, we are unable to specify the optimal routing architectures in the flow comprehensively due to the limitations of the VPR package. The following features of our routing architectures are sacrificed:

— We are unable to include the detail description of different wire styles used in the FPGA's. In the VPR input files, the only parameters we can describe the wires are the unit resistance and capacitance, but not the wire width and spacing. In our implementation, we describe the RC-wires with different pitches by scaling the RC values of wires. In addition, it is not possible to specify transmission lines, which are good for long links.
— Only limited topologies could be specified in the VPR inputs, which are less than those we generated for evaluation. Thus we can only use the most similar topology that is able to be specified as the substitution of the optimal one.
— Also, the tradeoff between energy and area due to different wire styles could not be reflected in the implementation, because we have to specify the same distribution of different wire styles in each row and column using VPR. Therefore, we are not able to adjust the wire styles according to the traffic.

We conduct the experiments using the 7 MCNC benchmarks and compare the energy obtained from mesh architecture and the proposed architecture—we do not use the word "optimal" here since we actually modify the optimal one due to the tool limitations we mentioned above. The results are presented in the first part of Table II.

The results indicate that the proposed architecture is superior to the traditional mesh in 5 cases out of 7, and the largest improvement reaches nearly 12%. Only in the alu4 case, it loses a relatively larger gap about 4.6%. In addition, we

Table II. Improvement of Energy of Proposed Architectures over Mesh Architecture

Benchmark	**Mesh Energy** ($x10^3 pJ$)	**Prop Energy** ($x10^3 pJ$)	**Imp** (%)
alu4	2.9909	3.1284	−4.597
apex4	1.8455	1.6256	11.916
diffeq	1.6265	1.6070	1.199
dsip	2.4504	2.4339	0.673
ex5p	2.0837	1.9674	5.581
misex3	2.7307	2.6156	4.215
tseng	1.7527	1.7670	−0.816
apex5	0.9616	0.9247	3.837
i8	1.1949	1.2014	−0.544
pair	1.5822	1.5627	1.233
pdc	7.9729	6.7312	15.574

Table III. Settings for Different Routing Architectures

	A	**B**
1. Transmission Line	No	Yes
2. Topology	Segmented	General
3. Wire Capacity	Fixed	Flexible

also using another 4 MCNC benchmarks as the "evaluation" benchmarks to test the goodness of the proposed architecture. The results are shown in the second part of Table II. It is observed that the performance is also satisfying: the energy is reduced in 3 out of 4 cases, and the largest one is 15.5%. In contrast, the only case whose energy is increased only loses 0.5%. Notice that we have sacrificed quite a few good features of the optimized routing architecture, according to the aforementioned explanations. To further show how the limitations of VPR affect the performance, we conduct the following experiments.

We first run our design flow on the representative netlist with $4000um$ routing area to obtain the optimize routing architecture, and evaluate the energy on each benchmark. We have different settings on the wire styles, topology and edge capacities, as shown in Table III. We consider three types of settings: without transmission line (1A) or with transmission line (1B); topology is either segmented; that is, each wire track has the same length, but just segmented to different portions (2A), or general topology is available (2B); the capacity of each wire style is fixed (3A), or it is flexible based on traffic patterns (3B). We then show the energy improvements over the mesh architecture with various settings in Figure 13.

The first setting (1A+2A+3A) is the one that could be specified using the VPR tool and we use for comparison in Table II. We can see that it could improve the energy by around 7% in average. With more options on topologies and flexible capacities of wires, the savings could be further improved. It is worth noting that the change of topology does not benefit much. That is because the segmented topology is similar to the optimal topology (Comparing with the mesh topology there is still a large gap). Finally, if transmission line is added, we could further reduce the energy by 10%. These comprehensive results indicate

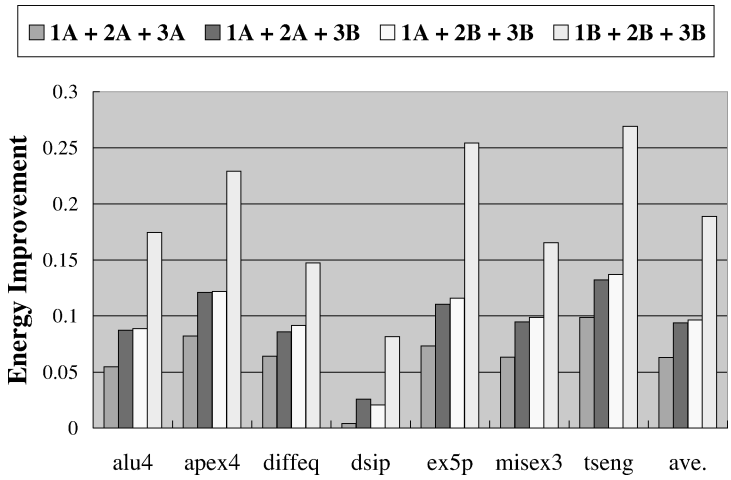

Fig. 13. Energy improvement with different settings over mesh architecture.

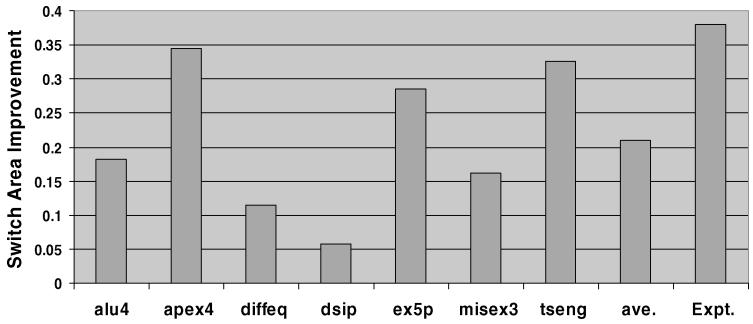

Fig. 14. Improvement of switch area of optimized architectures over mesh architecture.

that the different settings actually affect the performance of different architectures a lot, and the limitations of VPR lower the performance of the optimal architecture in the comparison in Table II.

5.2 Switch Area Optimization

Our methodology can be easily applied to various design objectives. In this experiment, we optimize switch area of FPGA routing architectures. We use the number of total switches in switch box as objective. Since the total number of switches is not effected by wire styles, the routing area constraint is not a major issue in switch area optimization. Figure 14 shows the switch area improvement when compared to mesh architecture for the seven benchmark circuits. In average, 15% to 20% switch area improvement can be seen.

5.3 Switch Area Constrained Energy Optimization

Furthermore, our methodology can combine energy and switch area optimizations in a unified optimization framework. In this experiment, we study the tradeoffs between them. Figure 15 depicts the optimized energy under various switch area constraints. The x-axis represents the number of switches in switch

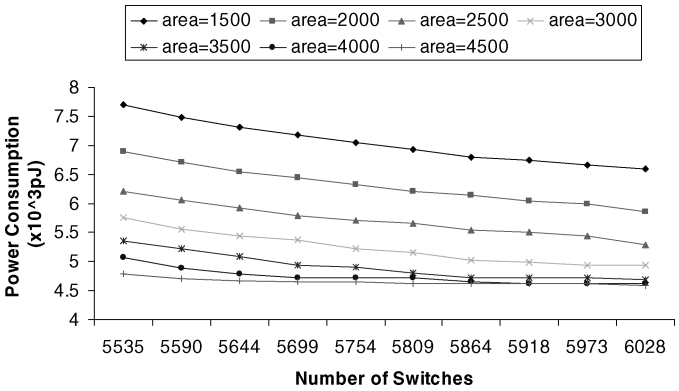

Fig. 15. Switch area constrained low power optimization for FPGA routing architectures.

boxes. The y-axis is energy in unit $\times 10^3 pJ$. Each curve represents the energy of the representative netlist under given routing area budget.

As the number of switches increases, energy decreases because the communication flow can be routed to more energy-efficient paths, which may have high switch costs. An interesting observation is that when the routing area budget is less, changing the switch area budget has larger impact on energy. For example, when changing number of switches from minimum to maximum, the energy changes by 16.7% for the curve area=1500um, which is only 4.6% for the curve area=4500um. This is because a tighter routing area budget necessitates the use of narrow but energy-costly wires, leaving a larger space for wire style optimization. When the routing area budget is abundant, the energy is already quite optimized no matter the switch area constraints.

6. CONCLUSIONS

In this article, we presented an improved MCF model based CAD flow that performs aggressive optimizations, such as topology and wire style optimizations, to reduce the energy and switch area of FPGA global routing architectures. The experiments show that when compared to traditional mesh architecture, our optimized architectures achieve up to 10% to 15% power savings and up to 20% switch area savings in average for a set of seven benchmark circuits. As future work, we can apply the methodology to other design objectives, such as interconnect delay in FPGA global routing architectures.

ACKNOWLEDGMENT

The project is partially supported by California MICRO Program. We thank Dr. Mike Hutton at Altera for his valuable discussion during this work.

REFERENCES

ALBRECHT, C. 2000. Provably good global routing by a new approximation algorithm for multicommodity flow. In *Proceedings of the International Symposium on Physical Design*. 19–25.

ALBRECHT, C., KAHNG, A., MANDOIU, I., AND ZELIKOVSKY, A. 2007. Multicommodity flow algorithms for buffered global routing. In *Approximation Algorithms and Metaheuristics*, Chapter 80.

BETZ, V. AND ROSE, J. 1997. Vpr: A new packing, placement and routing tool for FPGA research. In *Proceedings of International Workshop on Field-Programmable Logic and Applications*. 213–222.

BETZ, V., ROSE, J., AND MARQUARDT, A. 1999. *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers.

BROWN, S., KHELLAH, M., AND LEMIEUX, G. 1996. Segmented routing for speed-performance and routability in field-programmable gate arrays. *J. VLSI Des. 4,* 4, 275–291.

BROWN, S., KHELLAH, M., AND VRANESIC, Z. 1996. Minimizing fpga interconnect delays. *IEEE Des. Test Mag.* 16–23.

CARDEN, R. AND CHENG, C. 1991. A global router using an efficient approximate multicommodity multiterminal flow algorithm. In *Proceedings of the Design Automation Conference*. 316–321.

CARDEN, R., LI, J., AND CHENG, C. 1996. A global routing with a theoretical bound on the optimum solution. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst. 15*, 208–216.

CHEN, H., SHI, R., CHENG, C., AND HARRIS, D. 2005. Surfliner: A distortionless electrical signaling scheme for speed of light on-chip communications. In *Proceedings of the IEEE International Conference on Computer Design*. 497–502.

CHOW, P., SEO, S., ROSE, J., CHUNG, K., PAEZ, G., AND RAHARDJA, I. 1999. The design of an sram-based field-programmable gate array, part i: Architecture. *IEEE Trans. VLSI Syst. 7*, 191–197.

CHOY, N., CHIN, S., LEE, C., POON, K., LAMOUREUX, J., YAN, A., WILTON, S., ANG, S.-S., AND LUK, W. Power modeling for FPGA's. *http://www.ece.ubc.ca/stevew/powermodel.html*.

CHRISTIE, P. AND STROOBANDT, D. 2000. The interpretation and application of Rent's rule. *IEEE Trans. VLSI Syst. 8,* 6, 639–648.

CONG, J. AND DING, Y. 1994. Flowmap: An optimal technology mapping algorithm for delay optimization in lookup-table based fpga designs. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.* 1–12.

DAMBRE, J., STROOBANDT, D., AND CAMPENHOUT, J. V. 2004. Toward the accurate prediction of placement wire length distributions in vlsi circuits. *IEEE Trans. VLSI Syst. 12,* 4, 339–348.

DEHON, A. AND RUBIN, R. 2004. Design of fpga interconnect for multilevel metallization. *IEEE Transactions on VLSI Syst. 12,* 10, 1038–1050.

DONATH, W. E. 1981. Wire length distribution for placements of computer logic. *IBM J. Resear. Devel. 25*, 152–155.

GARG, N. AND KONEMANN, J. 1998. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *Proceedings of the 39th Annual IEEE Symp. on Foundations of Computer Science*, 300–309.

HU, Y., CHEN, H., ZHU, Y., CHIEN, A. A., AND CHENG, C. 2005. Physical synthesis of energy-efficient nocs through topology exploration and wire style optimization. In *Proceedings of the International Conference on Computer Design*. 111–118.

HU, Y., ZHU, Y., CHEN, H., GRAHAM, R., AND CHENG, C. 2006. Communication latency aware low power NOC synthesis. In *Proceedings of the Design Automation Conference*. 574–579.

HUTTON, M., ROSE, J., GROSSMAN, J., AND CORNEI, D. 1998. Characterization and parameterized generation of synthetic combinational benchmark circuits. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst. 17,* 10, 985–996.

KHELLAH, M., BROWN, S., AND VRANESIC, Z. 1994. Minimizing interconnection delays in array-bassed fpgas. In *Proceedings of the Custom Integrated Circuits Conference*. 181–184.

LEE, S., XIANG, H., WONG, D., AND SUN, R.Y. 2003. Wire type assignment for fpga routing. In *Proceedings of the International Symposium on Field Programmable Gate Arrays*. 61–67.

LEMIEUX, G., LEE, E., TOM, M., AND YU, A. 2004. Directional and single-driver wires in FPGA interconnect. In *Proceedings of the IEEE International Conference on Field-Programmable Technology*. 41–48.

MEHLHORN, K. 1988. A faster approximation algorithm for the steiner problem in graphs. *Inform. Process. Lett. 27,* 3, 125–128.

POON, K., WILTON, S., AND YAN, A. 2002. A detailed power model for field programmable gate arrays. *ACM Trans. Des. Automat. Electr. Syst. 10,* 2, 279–302.

SENTOVICH, E. AND AL, E. 1992. Sis: A system for sequential circuit analysis. Tech. rep. No. UCB/ERL M92/41, University of California, Berkeley.

SHAHROKHI, F. AND MATULA, D. 1990. The maximum concurrent flow problem. *J. Assoc. Comput. Math. 37,* 2, 318–334.

SHYU, M., WU, G., CHANG, Y., AND CHANG, Y. 2000. Generic universal switch blocks. *IEEE Trans. Comput. 49,* 4, 348–359.

YANG, S. 1991. Logic synthesis and optimization benchmarks, version 3.0. Tech. rep. Microelectronics Center of North Carolina.

YE, A. AND ROSE, J. 2005. Using bus-based connections to improve field-programmable gate array density for implementing datapath circuits. In *Proceedings of the International Symposium on Field Programmable Gate Arrays*. 3–13.

YE, A., ROSE, J., AND LEWIS, D. 2003. Architecture of datapath-oriented coarse-grain logic and routing for FPGAs. In *Proceedings of the Custom Integrated Circuits Conference*. 61–64.

Received October 2007; revised May 2008; accepted July 2008